2017-09

GPS enabled semi-autonomous robot

Bench, Connor F.

Monterey, California: Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

GPS ENABLED SEMI-AUTONOMOUS ROBOT

by

Connor F. Bench

September 2017

Thesis Advisor: Xiaoping Yun
Second Reader: James Calusdian

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 2017	**3. REPORT TYPE AND DATES COVERED** Master's thesis	
4. TITLE AND SUBTITLE GPS ENABLED SEMI-AUTONOMOUS ROBOT		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Connor F. Bench			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB number ____N/A____.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.		**12b. DISTRIBUTION CODE**	

13. ABSTRACT (maximum 200 words)

The primary objective of this research is to integrate GPS and local sensory data to allow a robot to operate semi-autonomously outside of a laboratory environment. The Pioneer 3-AT, a robust platform capable of operating in the outdoors, is utilized in this thesis. The P3-AT has acoustic sensors that can calculate distances to obstacles and encoders that calculate how much each wheel has turned. In a laboratory environment, sensory and encoder information can be used to triangulate position or measure distance and direction traveled from a known starting point. Operating outdoors limits the effectiveness of both systems as the obstacles are not known and wheels can often slip and slide on different surfaces. This necessitates external data to determine the location of the robot. GPS was chosen to provide that data. GPS, acoustic, and encoder data were integrated within MATLAB and provided control signals to the robot. The robot successfully navigated to a user-defined goal.

14. SUBJECT TERMS mobile robot, GPS navigation, potential field path planning.			**15. NUMBER OF PAGES** 65
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

GPS ENABLED SEMI-AUTONOMOUS ROBOT

Connor F. Bench
Lieutenant, United States Navy
B.S., United States Naval Academy, 2011

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2017

Approved by: Xiaoping Yun
Thesis Advisor

James Calusdian
Second Reader

R. Clark Robertson
Chair, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The primary objective of this research is to integrate GPS and local sensory data to allow a robot to operate semi-autonomously outside of a laboratory environment. The Pioneer 3-AT, a robust platform capable of operating in the outdoors, is utilized in this thesis. The P3-AT has acoustic sensors that can calculate distances to obstacles and encoders that calculate how much each wheel has turned. In a laboratory environment, sensory and encoder information can be used to triangulate position or measure distance and direction traveled from a known starting point. Operating outdoors limits the effectiveness of both systems as the obstacles are not known and wheels can often slip and slide on different surfaces. This necessitates external data to determine the location of the robot. GPS was chosen to provide that data. GPS, acoustic, and encoder data were integrated within MATLAB and provided control signals to the robot. The robot successfully navigated to a user-defined goal.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

GPS	Global Positioning System
KML	keyhole markup language
NMEA	National Marine Electronics Association
NNSS	Navy Navigation Satellite System
P3-AT	Pioneer 3-AT mobile robot built by Omron Adept MobileRobots
P3-DX	Pioneer 3-DX mobile robot built by Omron Adept MobileRobots

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank Professor Yun and Dr. Calusdian for their patience and guidance. Additionally, the students working in the Control Systems Laboratory were integral in maintaining a fun and challenging workspace, without which this project may not have been finished.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A platform that brings together navigational data from the U.S. government's Global Positioning System (GPS) and sensory data from a Pioneer 3-AT (P3-AT) mobile robot to provide control signals to the robot and allow it to operate outside of a laboratory environment is developed in this thesis.

There has been significant research into autonomous robotics in both military and civilian applications; however, this platform is designed to build off the coursework and, more specifically, the laboratory work of the Fundamentals of Robotics (EC4310) class taught at the Naval Postgraduate School [1]. The similarity between the platform developed in this thesis and the course work allows future students to move beyond the laboratory and into the field without having to learn a new coding language or become familiar with a new platform.

One of the core tenets of robotics is localization: the ability of a robot to determine its location and orientation within a reference frame [2]. Without knowing where it is in space, a robot is useless. It will be unable to navigate to a goal as it will not know in which direction to travel or even whether to stop should it stumble upon the goal by chance.

In wheeled robotic platforms, localization is conducted with encoders. Encoders measure how much each wheel turns. From a known starting point, a robot's position and orientation can be calculated through this measurement [3]; however, localization based on encoders is susceptible to error outside of a laboratory environment. In a natural environment, the robot's wheels roll across various different surfaces. Wheel slippage encountered when operating on gravel, dirt, grass, or other mediums introduces localization error. To operate effectively in such mediums, additional information is required.

GPS is a 24-hour-a-day, all-weather navigation system. It provides navigation information accurate to within a few meters. By determining the goal in GPS coordinates

and utilizing a GPS receiver (BU-353S4) onboard the robot, we see that the system can counter the encoder error and achieve accurate outdoor navigation.

On its own GPS is not enough. At slow speeds GPS has difficulty calculating directional change, and a refresh rate of one Hz leaves time between updates for the robot to move and not know where it is. By combining the two systems, the advantages of each are used to counter the other's weaknesses.

The design and testing of the control platform takes place in three steps. First, the GPS signal provided by the BU-353S4 is analyzed using a laptop computer as a test platform. The mobility of the laptop computer is a requirement as there is no GPS signal within the laboratory. Once the signal has been analyzed and a MATLAB platform has been developed to extract the specific information needed for this project from the signal as a whole, that platform is transferred to the SlimPRO SP675P microcomputer mounted on the P3-AT. The second step in the process is to use the SP575P to provide control inputs to the P3-AT and to demonstrate an ability to navigate to a goal defined in GPS coordinates within an environment free of obstacles. This requires comparing the user-defined goal and the robots current position to determine the distance and direction between the two. Once the robot is able to navigate to a goal, the third and final step is to implement potential field path planning to allow the robot to make decisions and avoid obstacles on its way to the goal.

The platform developed for this project provides a foundation for future research. Incorporating additional sensory components such as a magnetometer or laser range finder creates a robot capable of much more accurate navigation and obstacle avoidance. The continued development and interaction with this system will provide valuable experience to future Naval Postgraduate School students.

II. BACKGROUND

Fundamentals of Robotics is the entrance to robotics for students at the Naval Postgraduate School. The course utilizes a robot produced by Omron Adept MobileRobots, the P3-DX, to illustrate concepts and give students experience in a laboratory environment. The pertinent parts of that course are the concept of reference frames, the mathematics of moving between reference frames, encoders, and robot path planning with emphasis on the potential field method.

A. REFERENCE FRAMES

The foundation of robotics is the understanding of how objects interact in space. In order to describe different objects in a system, each object needs a location and an orientation in space. The location and orientation is defined with respect to a reference coordinate system or reference frame. In a two-dimensional plane {A}, any point can be described by its distance from the X-axis P_x and its distance from the Y-axis P_y. Combined they make the position vector ${}^{A}P$ as

$$ {}^{A}P = \begin{bmatrix} p_x \\ p_y \end{bmatrix} . \tag{1} $$

While this position vector gives the object a location, it does not describe its orientation. To describe the orientation of the object, we attach a coordinate system {B} to the object and describe it relative to the reference coordinate system {A} [2].

One way to do this is to write the unit vectors of the two principal axes $\hat{X}, \hat{Y}$ in {B} projected onto the axes of {A}. A rotation matrix ${}^{A}_{B}R$ describing the reference frame {B} in terms of {A} is created by combing the two unit vectors. An object viewed in the coordinate system {B} can then be transformed into the coordinate system {A} by utilizing the rotation matrix ${}^{A}_{B}R$, the position vector ${}^{B}P$, and the vector between the two frames ${}^{A}P_{BORG}$ as

$$ {}^{A}P = {}^{A}_{B}R\,{}^{B}P + {}^{A}P_{BORG} . \tag{2} $$

A point in one reference frame can be viewed within a different reference frame, as shown in Figure 1. To create a single operator to move a point in the {B} reference frame into the {A} reference frame, we define a matrix operator

$$\begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{A}_{B}R & {}^{A}P_{BORG} \\ 0 \quad 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix} . \tag{3}$$

The matrix operator is the transformation matrix and includes a rotational aspect ${}^{A}_{B}R$ and a translational aspect scaled by ${}^{A}P_{BORG}$ [2]. Equation (3) can be simplified as

$${}^{A}P = {}^{A}_{B}T \, {}^{B}P \ . \tag{4}$$

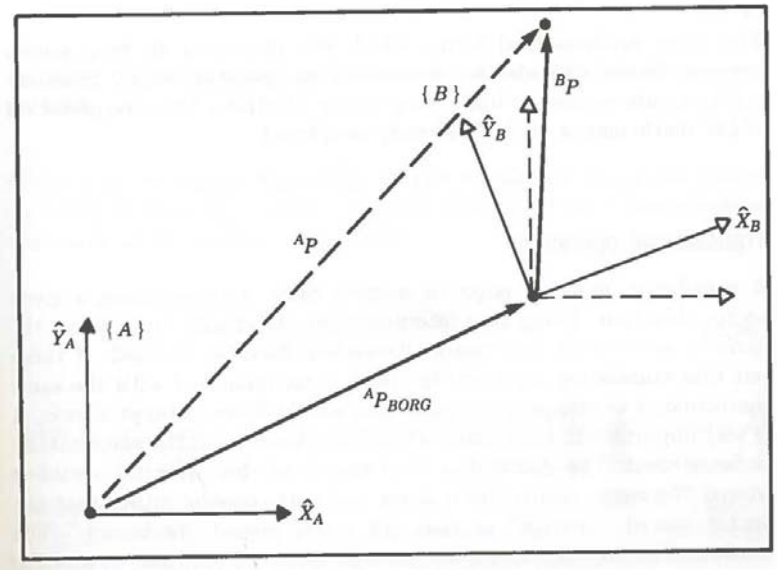

Figure 1. Point in {B} Mapped to {A}. Source: [2].

B. ENCODERS

The organic navigation system of the Pioneer P3-DX uses dead-reckoning (DR) to determine its location on an X-Y grid. The X-Y grid is organic to the robot and is redefined each time the robot is initialized. Upon initialization, the robot defines a reference frame with itself at the origin, the forward direction of the robot is defined as the Positive-X direction, and 90 degrees to the left is defined as the Positive-Y direction. The reference frame of the P3-DX, illustrated in Figure 2, is identical to the reference

frame of the P3-AT. The calculations used to determine its position are based on how much each wheel has turned. The wheel turn measurements are provided by encoders.

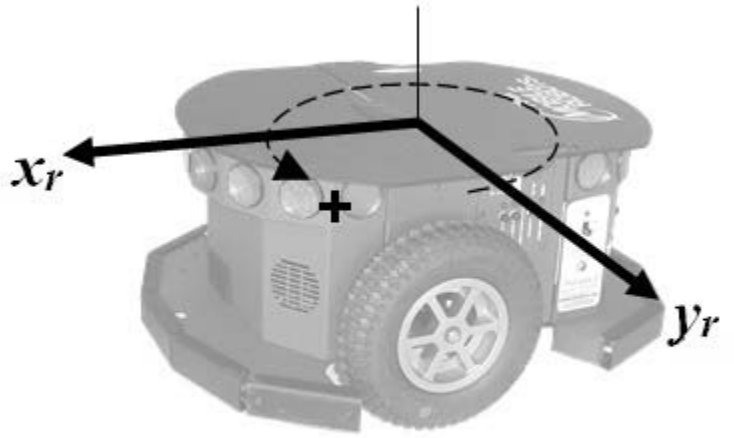

Figure 2. Reference Frame of P3-DX Mobile Robot. Source: [4].

Encoders operate in a variety of ways. The simplest and most popular is the optical encoder. The type of optical encoder used in the P3 robot series is an incremental encoder. The encoder disc has a number of evenly spaced gaps cut into the disc. A focused laser or other light source illuminates an area smaller than a single gap. As the disc rotates, the light shines through in areas where there is a gap and is blocked by the disc where there is no gap. When light shines through the gap, it hits a sensor. By counting the number of light pulses and measuring the time between pulses, we can determine how much the wheel has turned and at what velocity [3].

While the function and operation of the encoder(s) is somewhat transparent to the end user, encoder error plays a critical role in the design process and is explored further in a later chapter. This error is not in the measurement of the wheel rotation or velocity but rather in how accurately that data shows the movement of the robot as a whole.

C. PATH PLANNING

Path planning is a crucial portion of autonomous robotics. For the robot to reach the goal set by the user, it must have a way to determine where the goal is, the robot's current location, and a route between the two. Potential field path planning is utilized in this research.

There are two pieces of potential field path planning. The first is an attractive force F_{att} based on the location of the robot and its proximity to the assigned goal. The second portion is a repulsive force F_{rep} based on obstacles encountered in the robot's path toward the goal. These two forces are independent of each other. The artificial force is created by differentiating the potential function U as

$$\vec{F}(q) = -\vec{\nabla}U(q) \ , \tag{5}$$

where $\vec{\nabla}U(q)$ is the gradient vector of U at $q = (x, y)$, given by

$$\vec{\nabla}U = \begin{pmatrix} \dfrac{\partial U}{\partial x} \\ \dfrac{\partial U}{\partial y} \end{pmatrix} . \tag{6}$$

The total potential can be broken into an attractive potential U_{att} and a repulsive potential U_{rep} for any point q as

$$U(q) = U_{att}(q) + U_{rep}(q) \ . \tag{7}$$

This leads to the force $\vec{F}$ being the sum of two vectors: $\vec{F}_{att} = -\vec{\nabla}U_{att}$ and $\vec{F}_{rep} = -\vec{\nabla}U_{rep}$ [3].

A graphic depiction of the forces in potential field path planning is shown in Figure 3, where part (d) is a representation of the combination of the attractive potential (b) and the repulsive potential (c) [5].

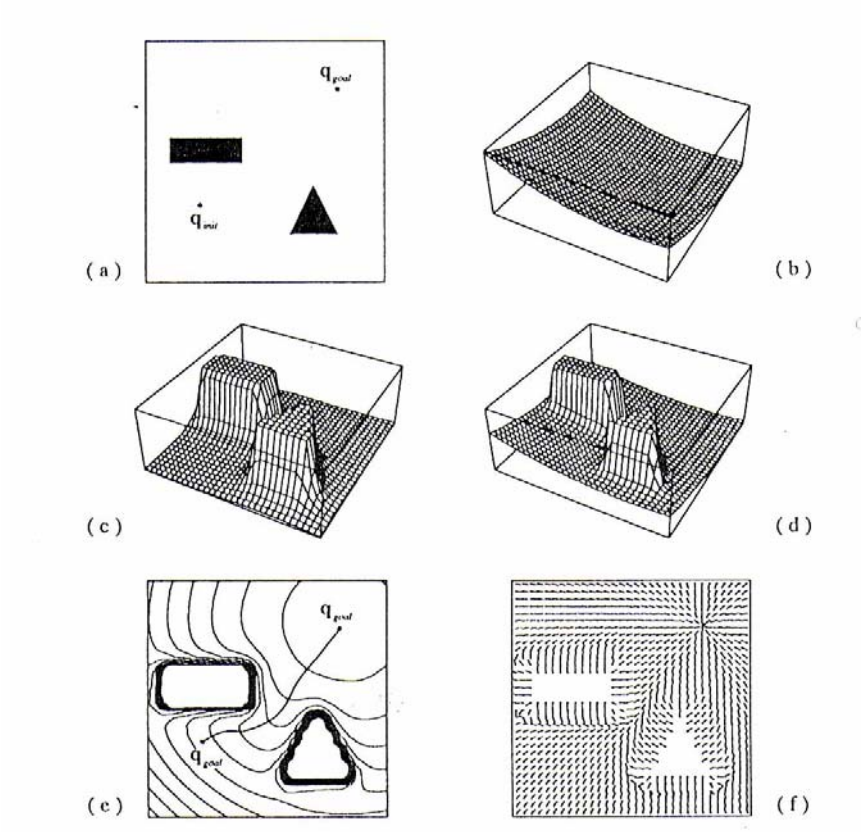

Figure 3. Graphic Representation of Potential Field Path Planning. Source: [5].

1. The Attractive Force

A simple method for calculating an attractive force is to define the attractive potential as a parabolic well

$$U_{att}(q) = \frac{1}{2}\xi\rho_{goal}^2(q) \ , \tag{8}$$

where ξ is a positive scaling factor and $\rho_{goal} = \left\| q - q_{goal} \right\|$ [5]. In this format, the attractive force is zero at the goal and increases as $\rho_{goal}(q) \to \infty$. The force is then determined by taking the derivative of the potential in Equation (8) giving

$$\vec{F}_{att}(q) = -\xi(q - q_{goal}) \ . \tag{9}$$

If, instead, the attractive potential is calculated as a conic well

$$U_{att}(q) = \xi\rho_{goal}(q) \ , \tag{10}$$

then the resultant force

$$\vec{F}_{att}(q) = -\xi(q - q_{goal}) / \left\| q - q_{goal} \right\| \tag{11}$$

has constant amplitude regardless of distance from the goal [4].

For this thesis, a combination of a parabolic well and a conic well was used to calculate the attractive force. While the robot was outside of a specified threshold ρ_0, the conic force was used, and as the robot approached the goal, the force was calculated via the parabolic well to take advantage of the stabilizing characteristic of the force approaching zero as the robot approached the goal [5].

2. The Repulsive Force

The intent of the repulsive force is to ensure the robot is able to avoid obstacles in its path. If an obstacle is sufficiently far away as to allow the robot to operate safely, it should have no impact on the robot's path. Once the robot crosses the safety threshold, ρ_0, the repulsive force of the obstacle increases to infinity as the distance between the robot and the obstacle approaches zero. The potential function

$$U_{rep}(q) = \begin{cases} \frac{1}{2}\eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)^2 & if\ \rho \le \rho_0, \\ 0 & if\ \rho > \rho_0, \end{cases} \tag{12}$$

gives the force function with the scaling factor η [5]. Taking the derivative of this repulsive potential leaves the repulsive force

$$\vec{F}_{rep} = \begin{cases} \eta\left(\frac{1}{\rho(q)} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2(q)}\vec{\nabla}\rho(q) & if\ \rho(q) \le \rho_0 \\ 0 & if\ \rho(q) > \rho_0 \end{cases}. \tag{13}$$

By combing the attractive and repulsive forces and following the resultant force vector, we find that the robot eventually arrives at its goal. The repulsive force can become complicated by multiple sensors or multiple obstacles. The techniques used to remedy this are covered in a later chapter.

III. GLOBAL POSITIONING SYSTEM

A. HISTORY

A map or chart is only useful if the current location of your ship or vehicle in relation to the mapped area is known. Accurately determining one's location is a critical portion of navigation. From celestial navigation to today's GPS, the quest for improved accuracy and precision has been never ending.

The DOD's first foray into satellite-based navigation was with the Navy Navigation Satellite System (NNSS), also known as TRANSIT, launched in 1959. It was a system made up of six satellites orbiting at altitudes of about 1100 km. With only six satellites in the system, a user could wait as much as 90 minutes for a satellite to pass overhead and provide location information. This was a significant drawback to the system and was part of the reason GPS was developed to replace it [6].

> The Navstar Global Positioning System is an all-weather, space based navigation system under development by the Department of Defense to satisfy the requirements for the military forces to accurately determine their position, velocity, and time in a common reference system, anywhere on or near the Earth on a continuous basis.—W.Wooden (1985) [6].

The GPS satellite constellation is made of 24, evenly spaced, satellites in circular orbits. This constellation design allows for a minimum of four satellites to be visible from any point on Earth at any time. This configuration is necessary to provide 24-hour-a-day instantaneous navigation information. The first GPS satellite was launched in 1978, and the system was fully operational in 1995 [6]. As new and improved GPS satellites have launched, the actual number of orbiting satellites has varied but never has fallen below the required 24.

B. SYSTEM COMPONENTS

There are three segments within GPS: a space segment consisting of satellites that broadcast signals, a control segment steering the entire system, and a user segment including a variety of receivers.

1. SPACE

The space segment is made up of the 24 broadcasting satellites orbiting the Earth. Each satellite is continuously broadcasting spread spectrum signals that are utilized by the user to determine navigation information. At least 24 satellites are required so that at any given time there are a minimum of four satellites visible to a user anywhere on the Earth.

2. CONTROL

The control segment is made up of a master control station, monitor stations, and ground control stations located across the globe. The segment's main role is tracking the satellites and monitoring the broadcast signals.

From the different monitoring stations, each satellite in the constellation is observed. These monitoring stations are located all over the world, as shown in Figure 4, to ensure that satellite observation is constant and accurate. Should a satellite be observed to deviate from its expected orbit, a control signal is sent to the satellite to update its navigation message broadcast to account for the updated position of the satellite in space [6].

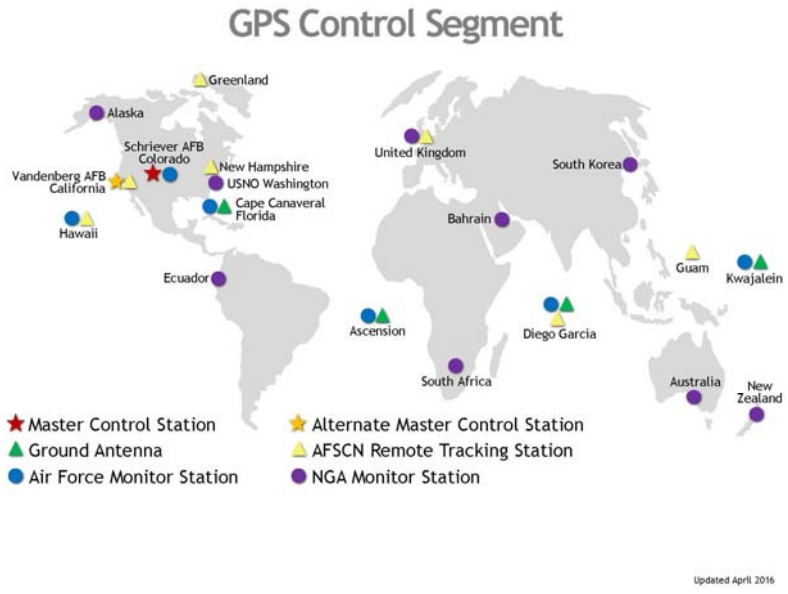

Figure 4. Map of GPS Control Segment. Source: [7].

3. USER

The user segment is made up of the military and civilian receivers that utilize GPS broadcast signals and determine user locations and velocities. The basic design of a GPS receiver is depicted in Figure 5. The antenna receives a signal from four or more satellites. The microprocessor compares those signals to the data it receives from the control device and returns navigational information to the control device to output to the user. Information is also stored depending on how the GPS receiver is being utilized.

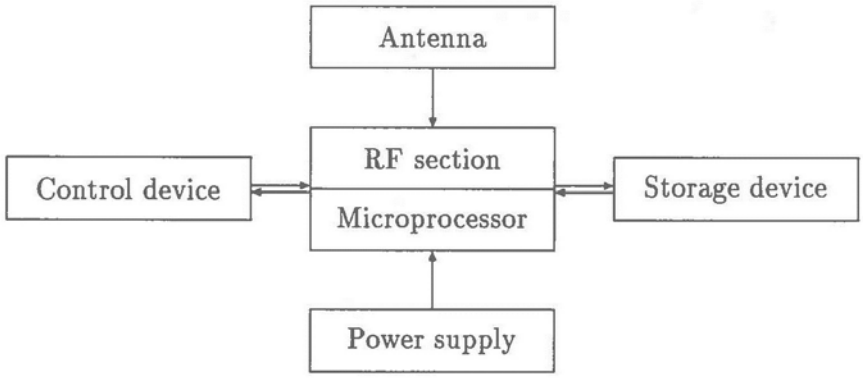

Figure 5. Basic Concept of a Receiver Unit. Source: [6].

C. SIGNAL

The data sent in a GPS signal consists of the satellites location and a time stamp. The receiver compares the time stamp of the signal with the time of its own clock to determine the amount of time the signal was in transit. By knowing the location of the satellite and the amount of time required for the signal to arrive, we see that the receiver can calculate a range. With three satellite signals, and thus three spherical ranges from known points, the receiver calculates its exact location. A fourth satellite is required to account for error in the receiver's clock. While each GPS satellite is equipped with an exceptionally accurate atomic clock, typical receivers utilize cheap and error-prone quartz clocks. The clock error of the receiver constitutes a fourth unknown. The distance R to a given satellite is calculated from a range q and a range correction Δq as

$$R = q + \Delta q = q + c\delta \,. \tag{14}$$

The range correction is the clock error δ of the receiver multiplied by the speed of light c. Using Equation (14), we calculate the distances to three satellites R_1, R_2, and R_3. With δ, we now have four unkowns requiring four equations to solve, and there lies the requirement for four visible satellites [6].

The data provided by the GPS signals to the receivers is outputted to the user in a variety of protocols. The National Marine Electronics Association (NMEA) defines one such protocol that is used on the BU-353S4 GPS receiver used in this research. The NMEA protocol parcels the GPS data into sentences, two of which were used in this research. The $GPGLL sentence provides longitude and latitude information to the user and the $GPVTG sentence provides heading and velocity information [8]. These two data sentences will be discussed in detail in Chapter IV.

IV. DESIGN METHODS

The design process has a number of steps. The robot must be able to determine its location and orientation, the direction and distance to the goal, and any obstacles that may lie between the two. Data must be taken from the user to define the goal, from the GPS receiver to define the robot's position, and from the acoustic sensors organic to the P3-AT robot to observe obstacles in its path. The data from these sources is processed, and the output control signals are sent to the robot to direct its travel. MATLAB was chosen as the programming language due to its familiarity to all electrical engineering students and to facilitate follow on research.

A. GPS

Both the robot localization and the goal are defined with GPS coordinates. This allows for easy comparison. The distance between the robot's position and the goal constitutes the error the navigation system is attempting to minimize.

1. National Marine Electronics Association

The BU-353S4 GPS receiver outputs its data to the user via National Marine Electronics Association (NMEA) protocol. This means the data is outputted in a series of sentences, each one identified by a specific initial character string. The protocol covers a variety of marine electronics and how they communicate. The initial identifying character string of each data sentences consists of a dollar sign $ followed by five characters. The first two characters define the "talker" (i.e., what piece of electronic equipment is sending the signal), which for this project is always GPS or "GP." Characters three, four, and five identify the specific sentence being sent [8]. While there are over 50 different sentences defined within the NMEA protocol, this project focused on only two: GPGLL and GPVTG.

The GPGLL sentence, illustrated in Figure 6, provides the location of the robot in real time via latitude and longitude coordinates. The accuracy of the GPS receiver is down to the 1/100 of a minute (in the form degrees, minutes, seconds).

GLL Geographic Position – Latitude/Longitude

```
       1       2 3        4 5         6 7
       |       | |        | |         | |
$--GLL,llll.ll,a,yyyyy.yy,a,hhmmss.ss,A*hh

1) Latitude
2) N or S (North or South)
3) Longitude
4) E or W (East or West)
5) Time (UTC)
6) Status A - Data Valid, V - Data Invalid
7) Checksum
```

Figure 6. Breakdown of GPGLL Sentence. Source: [8].

The GPS information is pulled into MATLAB from the input buffer using the string compare command *strcmp*. This command looks at each line from the input buffer and compares it against a user-defined character string—in this case, the two sentence identifiers $GPGLL and $GPVTG. Once the program sees that string, it pulls the sentence out of the buffer and stores it in memory. The GPS data is then manipulated into decimal form for ease of future calculations.

2. Keyhole Markup Language

Once the robot has been localized, its position needs to be compared to the user-inputted goal. Initially, the goal was inputted directly into MATLAB by the user for simplification. To improve the user interface, we looked to Google Earth as a program to define the goal.

When using Google Earth, the program calculates the GPS coordinates of the location identified on the frame. This location can be defined as the center of the pictured area, as illustrated in Figure 7 at the bottom of the screenshot. Alternatively, the calculated location can be defined by the cursors location over the frame. Using a keyboard shortcut, we can copy that location from the keyhole markup language (KML) to the clipboard as a character string. That string can then be pulled from the clipboard

into MATLAB, redefined as a double-data type, and manipulated into the same decimal form as the GPS signal so they can be compared.

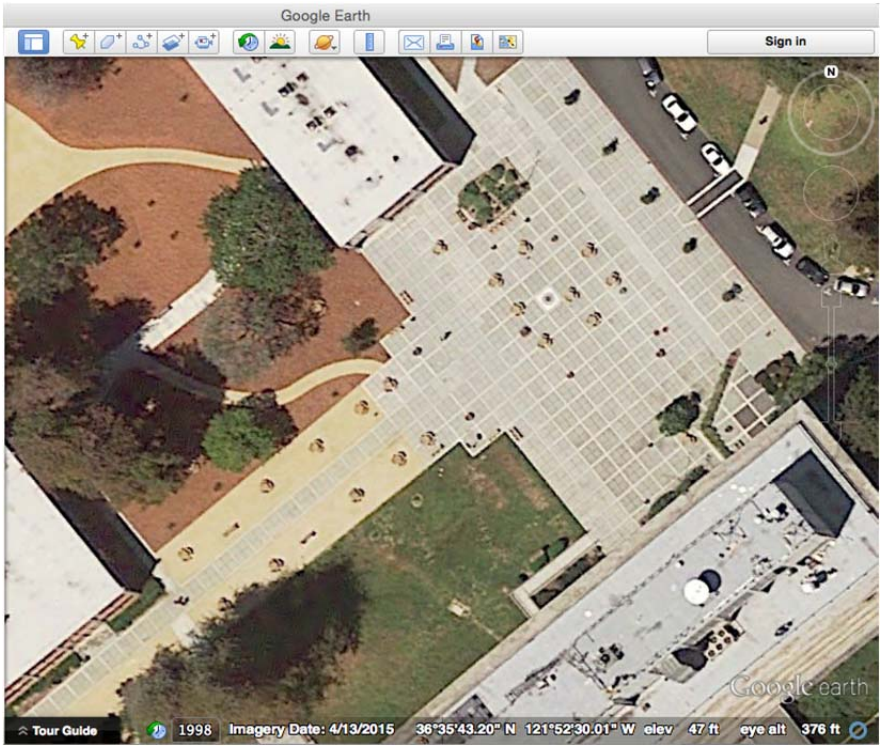

Figure 7. Google Earth Screenshot of Spanagel Courtyard

3. Haversine Formulas

The haversine formulas make up the calculations to determine distance and direction between two points on the Earth's surface defined in GPS coordinates. Distance is calculated by finding the angular distance between the two points on the surface of the earth and multiplying it by the radius of the Earth [9]. The average radius of the Earth $R_E = 6.371 \times 10^6$ m was used in this case for simplicity. Latitudes are defined as φ and longitudes as λ . First, we calculate the square of half the chord length between the two points

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos(\varphi_1)\cos(\varphi_2)\sin^2\left(\frac{\Delta\lambda}{2}\right). \quad (15)$$

We use Equation (15) and the *atan2* function to determine the angular distance in radians

$$k = 2\text{atan2}\left(\sqrt{a}, \sqrt{1-a}\right) , \tag{16}$$

which can then be used to determine the distance by multiplying by the radius of the Earth

$$d = R_E k \ . \tag{17}$$

The distance calculation works well even for very small distances. The bearing calculation, however, is much better suited for shorter distances. This is due to great circle navigation and the fact that the initial bearing can be very different from the final bearing even if traveling in a straight line. An example would be air travel in the northern hemisphere: often the shortest route between two cities involves flying over the arctic; in this case the aircraft is initially travelling north, but the second portion of the trip consists of traveling south. While the aircraft is constantly traveling in a straight line, the heading can change significantly over the course of the trip. The initial direction of travel θ is also calculated from the latitude and longitude of the robot and its goal using the atan2 function [9]

$$\theta = \text{atan2}\left[\sin(\Delta\lambda)\cos(\varphi_2), \cos(\varphi_1)\sin(\varphi_2) - \sin(\varphi_1)\cos(\varphi_2)\cos(\Delta\lambda)\right] . \tag{18}$$

The complication of a changing heading is not a factor as the length of travel in this application is on the order of m vice km.

4. Heading

Just as knowing the goal location is only useful if the present location is known, knowing the needed direction of travel is only useful if the current direction of travel is known. The GPS receiver provides the heading on which the robot is traveling. As with the GPGLL sentence, there is a specific sentence within the NMEA protocol that provides direction and velocity data. The components of the GPVTG sentence are broken down in Figure 8.

VTG Track Made Good and Ground Speed

```
       1   2 3   4 5  6 7   8 9
       |   | |   | |  | |   | |
$--VTG,x.x,T,x.x,M,x.x,N,x.x,K*hh
```

1) Track Degrees
2) T = True
3) Track Degrees
4) M = Magnetic
5) Speed Knots
6) N = Knots
7) Speed Kilometers Per Hour
8) K = Kilometres Per Hour
9) Checksum

Figure 8. Breakdown of GPVTG Sentence. Source: [8].

The MATLAB program strips out the true-track direction vice the magnetic. This allows the program to calculate the direction to the GPS goal in the robot reference frame by comparing the direction determined by the haversine function and comparing it to the heading on which the robot is currently moving. By knowing the direction and distance to the goal in the robot reference frame, we can move the GPS goal's coordinates to the robot's fixed reference frame and utilize them in potential field path planning.

B. ENCODER-GPS INTEGRATION

The encoders organic to the P3-AT are quite good at accurately measuring how much each wheel has turned; however, depending on the properties of the environment in which the robot is operating, those measurements may not be able to accurately indicate how far the robot has actually traveled. Encoder error accumulates over the robot's course of travel; however, if the system can be regularly updated, the accuracy of the wheel turn measurements can be useful and the error to the system can be mitigated.

Each time the P3-AT is initialized, it creates an organic grid system with forward being the positive X-direction and the positive Y-direction being 90 degrees to the left. This reference frame was named the "robot's fixed frame" as it is the foundation of the navigation system. As the robot is moving within the fixed frame, it maintains its immediate reference frame as described generally in Chapter II and specifically in Figure 1.

The simplest way to build the navigation system needed to operate outside the laboratory is to bring the GPS data into the organic grid system. This is done by using the transformation matrix described in Chapter II to bring the goal from the robot reference frame into the fixed frame. By running this calculation each time through the control loop, we are able to account for encoder error. The error is found in how far we have traveled from the known starting position, while what we care about is how much farther we must travel toward the known goal.

The encoders proved especially useful in calculating heading. For the GPS receiver to accurately output a heading and velocity, the robot had to be traveling at a speed greater than 1.2 m/s. While this does not sound particularly fast, to have no heading output when moving slowly in rougher terrain or maneuvering around obstacles makes it quite difficult for the system to provide accurate control signals. The heading values provided by the GPS are stored by MATLAB, and each time through the control loop the last two headings outputted from GPS are compared. If they are identical, the assumption it that the robot is traveling too slowly to update the headings, in which case the headings are calculated by the encoders instead.

C. ACOUSTIC DATA

The P3-AT is outfitted with 16 acoustic sensors. These sensors cover 360 degrees and have an effective range of 5.0 m. The sensors' locations and angle of direction are given in Table 1. The data provided by the sensors is a range to the closest obstacle along a straight path in the angle at which the sensor is pointed and starting from the defined location on the robot.

Table 1. Angles (degrees) and Locations (mm from center of robot) of P3-AT Acoustic Sensors. Source: [4].

Angle	X Dist (mm)	Y Dist (mm)
90	69	136
50	114	119
30	148	78
10	166	27
-10	166	-27
-30	148	-78
-50	114	-119
-90	69	-136
-90	-157	-136
-130	-203	-119
-150	-237	-78
-170	-255	-27
170	-255	27
150	-237	78
130	-203	119
90	-157	136

If there is nothing within 5.0 m of the sensor, it returns an output of 5.0 m. The difference between the measured value and the maximum value equates to the distance to an obstacle. The larger the difference between the two values, the closer the obstacle is to the robot.

Each sensor provides its data to the system. The aggregate data is used to determine the control output to the robot. It makes no difference if the sensors are seeing one big object or many smaller objects. The data each sensor provides is broken into its components. Using trigonometry and the angle at which each sensor is set, we calculate values for objects seen in each of the positive and negative X and Y directions, respectively.

D. POTENTIAL FIELD PATH PLANNING

The potential field method of mobile robot path planning can be described as the composite value of the GPS and acoustic sensor inputs. The goal input and obstacle inputs are each broken into their *X* and *Y* components. The goal input components are weighted and used to calculate an attractive force. The obstacle input components are summed and weighted to create a repulsive force. The attractive and repulsive forces are then compared to create the translational and rotational control outputs to the robot.

Illustrated in Figure 9, a local minimum occurs when the attractive and repulsive forces exactly cancel. The resulting configuration allows for no control inputs to effectively move the robot. This is a significant weakness of the potential field path planning method. The larger the concavity in which the robot reaches a local minimum the more difficult it is to remedy. One method is to identify any situation in which the attractive and repulsive forces are equal and the goal has not yet been reached (i.e., any time the robot has reached a local minimum), and direct the robot to travel in a specific direction a predetermined distance before returning to the potential path planning method. This method is effective if the C shaped obstacle is small enough, or if the predetermined distance traveled is great enough; however, it is not uncommon for the robot to fall back into the local minimum after attempting to move beyond it [5]. Other methods of combatting local minimums are significantly more complicated.

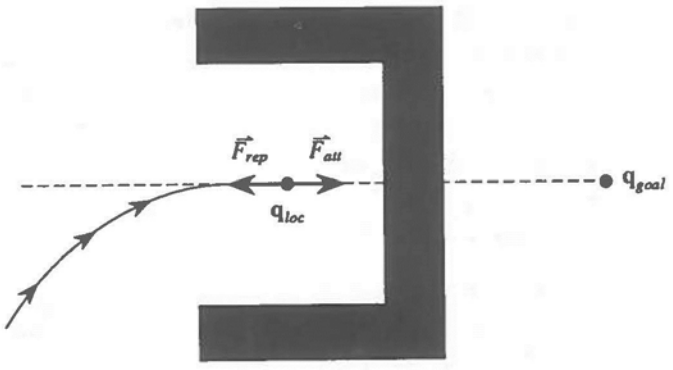

Figure 9. Local Minimum in Potential Field Path Planning. Source: [5].

The potential field path planning method is the foundation of the navigation system developed for the P3-AT robot. The distance and direction calculated by the haversine function using the GPS data inputted by the user and provided by the GPS receiver provides the attractive force. The repulsive force is determined by the data provided by the 16 acoustic sensors covering a 360-degree field-of-view. The comparison of the two forces provides an overall force that leads to a control input. The system uses a proportional controller with a translational velocity and a rotational velocity. These velocities are determined by the calculated attractive and repulsive forces and user-determined gains.

THIS PAGE INTENTIONALLY LEFT BLANK

V. TEST PROCEDURES

Testing occurred in three major parts during the design process. Initially, testing consisted of bringing the GPS signal into the MATLAB workspace and getting it into a form that could be more easily utilized within the control system. Later, those control signals were tested by maneuvering the robot to a user-defined GPS goal on a flat surface void of obstacles. Finally, the control signals were routed through the potential field path planning method to enable the robot to operate safely and avoid obstacles on a variety of mediums. Data was collected over a series of three runs that covered approximately 10 m of flat concrete and 10 m of uneven grass and dirt. This data is analyzed in Chapter VI.

A. INITIAL GPS ANALYSIS

The Naval Postgraduate School Control Systems Laboratory (SP-521) does not receive any GPS signals. In order to interact with GPS and receive accurate navigation data, the user must be outdoors and be within sight of at least four GPS satellites at all times. To that end, the most effective test method is to simply walk around outdoors with a laptop computer and the BU-353S4 GPS receiver. The SlimPRO SP675P microcomputer that is mounted on the robot does not have a monitor; the user must utilize a separate monitor or remote login to access it. While both methods are used in the final testing and in the system itself, using a laptop computer allows us to bypass those challenges for the initial signal analysis. The initial tests were designed to ensure accurate and timely GPS information could be moved from the input buffer, into MATLAB, and manipulated into a form that allows for easy calculation and comparison. These initial tests revealed two significant issues. First, the input buffer was too large, and second, the initial design for determining heading was insufficient.

The large input buffer meant the data pulled into MATLAB was not sufficiently close to real time. While any input buffer ensures data is not immediate, the size of the default input buffer was such that there was a delay of approximately 5.0 s. This means the control inputs sent to the robot were determined by the robot's location and sensory data from 5.0 s prior. Even at the slow speeds at which the robot travels the 5.0 s delay

was significant. The appropriate buffer size was determined by shrinking the input buffer incrementally, starting with the default size of 512 kb, to find a buffer size that allowed the system to function properly. The final size of 128 kb ensures the information in the buffer is close enough to real time to allow for effective control signal calculation. The final buffer size allows for an approximately 1.0 s delay between real time and data analysis.

The initial design of the platform solely utilized the GPGLL sentence. A number of methods were tried to determine the GPS receiver's heading from the location data. The basic concept was if the last two data points were known, calculating the direction travelled between them would be equivalent to, or at least a good approximation of, the direction of travel for the robot. In practice, this calculation resulted in inaccurate and inconsistent heading outputs due to the error intrinsic to the GPS system itself. The GPS system is designed with an accuracy of less than 6.0 m 95% of the time [10]. While the system consistently operates at accuracies of less than 2.0 m, utilizing the location data to determine heading requires filtering the data. Further research into the NMEA protocol identified the GPVTG data sentence, which provides heading and speed data directly [6]. In essence, the GPS receiver filters the data itself and provides a heading output. Once the GPVTG sentence was identified, providing the pertinent information to MATLAB was accomplished in much the same way as the GPGLL information.

B. STRICTLY GPS NAVIGATION

Once the location and heading data could be inputted into MATLAB, the next phase of the tests consisted of having the P3-AT travel to and stop at a user-defined GPS goal. This required utilizing the robot-mounted SlimPro microcomputer; which in turn required the use of a separate monitor and remote login.

The test station utilized, displayed in Figure 10, is comprised of a monitor, keyboard, and mouse directly plugged into the SlimPro microcomputer and a Microsoft SurfacePro that is used to remotely login to the microcomputer through a wireless network. The wireless network consists of a TL-R402M router manufactured by TP-LINK and a N150 wireless access point manufactured by NETGEAR. Plugging directly

into the microcomputer with a full size keyboard and monitor makes for easy data analysis and system troubleshooting. The wireless network and remote login allows the robot to operate untethered during testing.

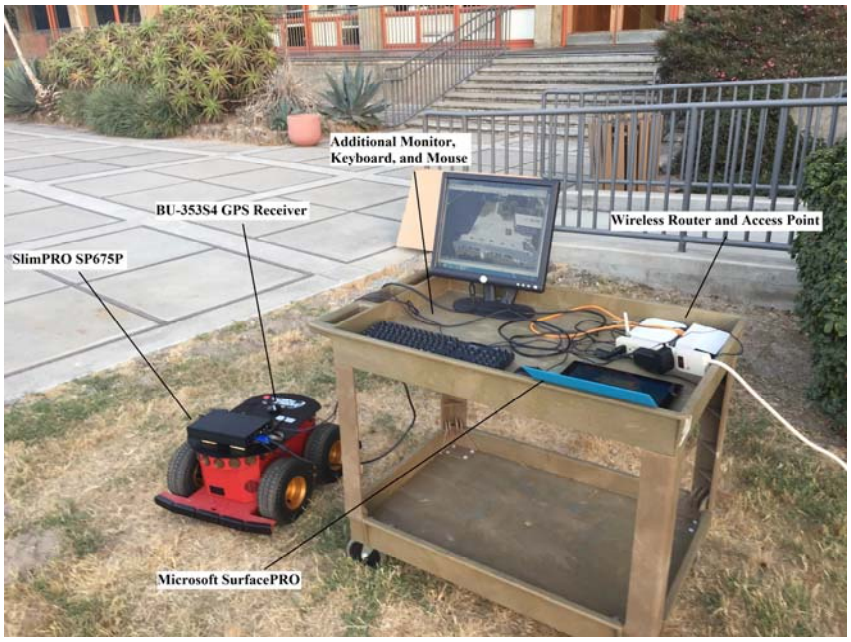

Figure 10. Test Station

The tests consisted of directing the robot to travel to a GPS goal from its original location. The navigation system used a proportional controller with distance to the goal determining translational velocity and the difference between the robot's current heading and the required heading determining the robot's rotational velocity. The path of the robot was constrained to flat, dry concrete, and the environment was devoid of obstacles. Consequently, the acoustic data was not utilized during this part of the project.

Accurately determining the direction of travel again proved difficult. The heading provided by the GPS receiver would cease regularly updating, and the GPVTG sentence provided the same heading with each cycle regardless of whether the robot was turning or not. The challenge is overcome by ensuring the robot travels at its maximum speed at all times. Further research into the specific BU-353S4 receiver determined the receiver must travel above a minimum threshold of 1.2 m/s for an accurate and updated heading output

to be produced [11]. The robot is capable of operating above that threshold, and in this particular test environment, a flat dry concrete area without obstacles, there is no problem. As the objective is to create a platform capable of operating in a diverse and unknown environment, the system needs further development.

C. COMPLETE SYSTEM TESTING

The initial system tests proved the GPS receiver capable of providing real-time and accurate navigational data to the robot. The final system needs to be able to take that data and incorporate it with acoustic sensory data and encoder data to determine appropriate control signals to ensure the robot is able to safely and efficiently arrive at its intended destination.

To avoid obstacles effectively, it is often necessary for the robot to slow down; however, once the robot's velocity drops below the 1.2 m/s threshold, the GPS receiver is unable to accurately update the robot's heading. While an inaccurate heading in and of itself only leads to inefficiency, an inability to update the heading results in the robot turning perpetually in a circle rather than proceeding to the goal. This is where an integration of the GPS data and encoder data becomes crucial.

An accurate heading is required to calculate the location of the GPS goal within the robot's reference frame. To ensure an accurate and updated heading at slow speeds, the system must rely on the encoders organic to the system. The heading measurement is initialized using the GPS data as the robot's initial heading is always 000 within the fixed frame. The difference between the initial GPS heading and 000 is calculated and stored. As the robot moves, the heading value within the robot's fixed reference frame is constantly updated via the encoder data. Should the robot slow and, thus, lose GPS data, the current fixed frame heading and the initial difference can be used to calculate the current heading in the GPS reference frame.

The integration of the acoustic data into the system and bringing all the inputs into the potential field path planning method was a fairly smooth process as much of the troubleshooting had already been encountered in a previous robotics class.

VI. RESULTS

All testing took place in the space between Spanagel Hall and Root Hall on the Naval Postgraduate School campus, as illustrated in Figure 11. The space consists of a grassy area, a wood chip-covered area, and a concrete area with numerous planting pots, light posts, benches, and tables located throughout. The robot was able to successfully navigate from a variety of starting points to any goal. The obstacle avoidance and potential field path planning method was tested extensively prior to this research and functioned as expected in the testing environment. The collection of test data and analysis focused on the integration of the GPS and encoder data.

Figure 11. Test Route in Spanagel Courtyard

Three test runs were conducted to a fixed user-defined GPS goal from a single starting point. The area between the starting point and the goal consisted of approximately 10 m of flat dry concrete and 10 m of wet, grass-covered ground.

Over the course of each run, the robot successfully navigated toward the goal, and the measured distance to the goal decreased with each control loop iteration, as shown in

Figure 12. The measured distance does not reach zero as the controller directs the robot to stop once it is within 2.0 m of the user-defined goal. The final distance to goal for each run was 1.93 meters, 1.75 meters, and 1.96 meters for Run 1, Run 2, and Run 3, respectively.

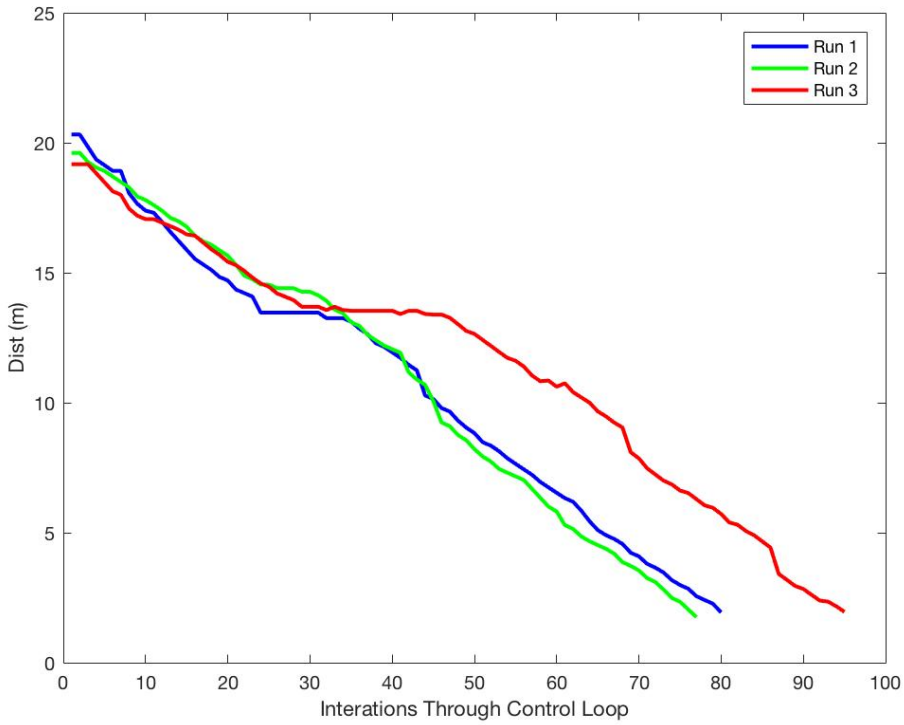

Figure 12. GPS Data Showing Distance to the Goal vs. Control Loop Iterations for Trial Runs 1, 2, and 3

The transition to grass for each test came at approximately 13 m from the goal. At the transition point there was an elevation change, and the raised grass appeared as an obstacle to the robot even though the gradient was small enough for the robot to navigate. In each instance, the robot identified the obstacle and turned to the right. Once the grass was not directly in front of it, the robot proceeded forward. As the front, left wheel began to climb the gradient, the attitude of the robot shifted, and the grass no longer presented as an obstacle. This allowed the robot to proceed over the hill and continue toward the

goal. This instance is depicted in Figure 12, where the magnitude of each slope decreases for a number of iterations of the control loop at approximately 13 m from the goal.

To account for encoder error, the GPS goal location in the robot's fixed reference frame was recalculated each time through the control loop. If the encoder data allows for the robot to appear to have travelled farther than it actually has, the location of the goal within the reference frame shifts the next time through the control loop to account for that error. The robot's path and the calculated positons of the goal for Runs 1 and 2 in the fixed reference frame are displayed in Figure 13.

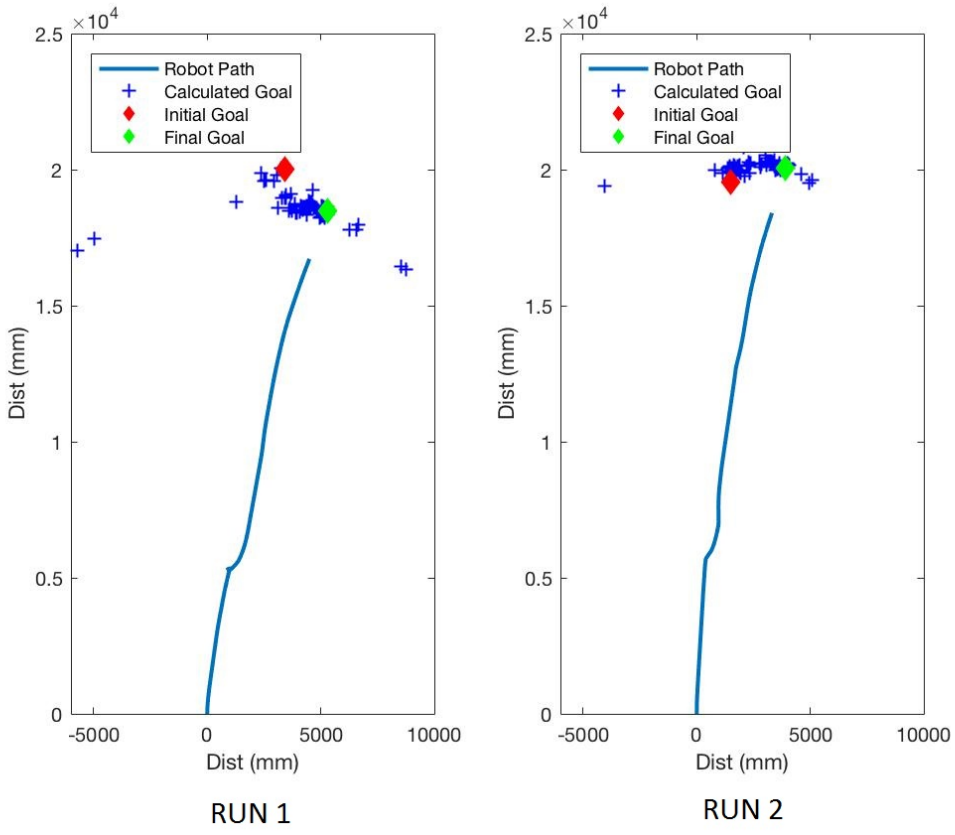

Figure 13. Robot Path and Calculated Goal Positions for Runs 1 and 2

The distances between the initial, calculated goal and the final goal positions are 1.57 m and 1.21 m for Run 1 and 2, respectively. This change is representative of error in the encoder data. The arc of calculated goal positions indicates that the majority of the

encoder error occurs during rotational movement vice translational movement. The physical location of the goal is not changing; rather, it is the robot's perception of the goal location that is changing.

The path and calculated goal locations for Run 3 are shown in Figure 14. The robot took a slightly different path for Run 3 as it travelled further to the right during the concrete to grass transition than it did for Runs 1 and 2. This made its path over the grass bumpier and explains the increase in translational error. The arc between the initial and final goals is still indicative of rotational error. The distance between the initial and final calculated goals for Run 3 was 8.05 m.

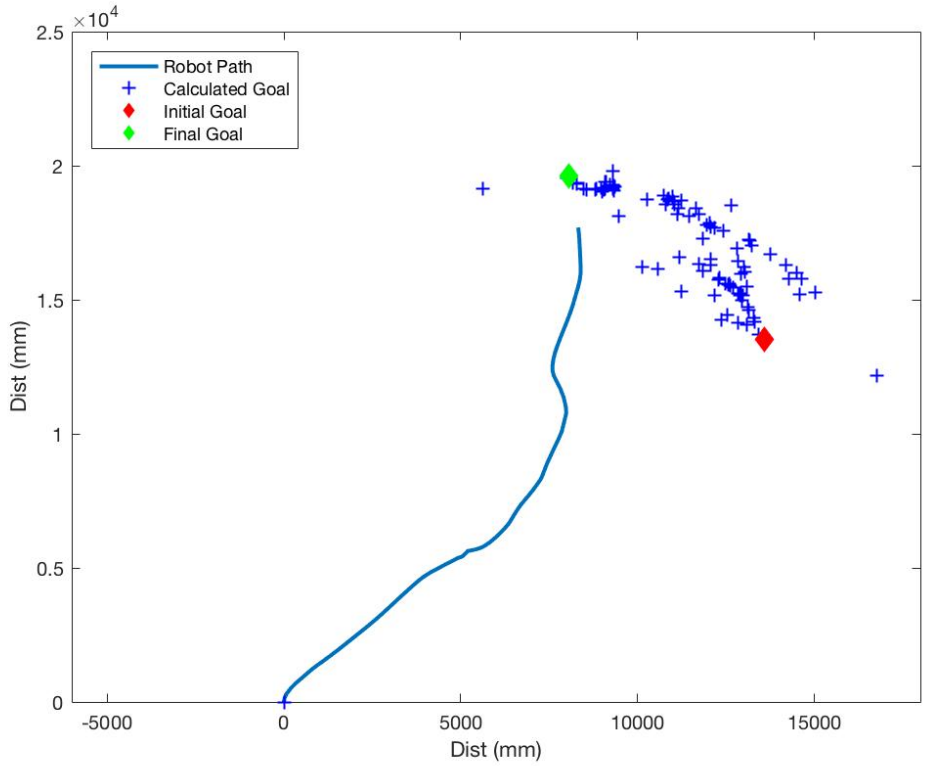

Figure 14. Robot Path and Calculated Goal Positions for Run 3

Calculating the distance to goal at the termination of the control loop within the fixed reference frame gives the same distances (1.93 m, 1.75 m, and 1.96 m) as the GPS data, indicating the transfer of data between the GPS reference frame and the robot's fixed reference frame was done correctly.

VII. CONCLUSIONS

A. SUMMARY

The objective of this research was to develop a MATLAB platform that integrates GPS, acoustic, and encoder data to avoid obstacles and navigate to a user-defined goal. The GPS data was integrated with the encoder data to accurately determine the robot's location. By comparing the robot's location to the user-defined goal, we determined an attractive force. A repulsive force was calculated from acoustic sensory data of obstacles along the robot's path. Summing the two forces and using the potential field path planning method allowed the robot to successfully navigate to the goal. This was demonstrated over three trail runs. Each run was made along a course consisting of approximately 10.0 m of flat, dry concrete and 10.0 m of wet, bumpy grass and terminated once the robot was within 2.0 m of the goal.

The data collected during the three trial runs demonstrates the robot's ability to successfully navigate from a starting point to a user-defined goal. In each instance, the navigation platform accurately took data from the GPS receiver, integrated it with acoustic and encoder sensory data, and directed the robot to within 2.0 m of the GPS goal.

The data imported from the BU-353S4 GPS receiver was compared to the user-defined GPS goal coordinates. Comparing the heading of the robot and the distance and direction to the GPS goal allowed the goal coordinates to be calculated in the robot's reference frame and then transferred into the fixed reference frame. An attractive force was calculated based on the location of the goal in the fixed reference frame. Acoustic sensory data covering 360 degrees allowed for calculation of a repulsive force for each obstacle observed within 5.0 m of the robot. The potential field path planning method implies that each time through the control loop the two forces are summed, and the resultant force directed the motion of the robot. The integration of encoder and GPS data allowed the robot to operate over grass and wood chips in addition to concrete.

The MATLAB navigation platform developed during this project provides control signals to allow the P3-AT robot to successfully operate outside of a laboratory environment.

B. FUTURE WORK

There is opportunity for future work to improve and add to this MATLAB navigation platform. The three test runs demonstrated that the greatest source of error in the system is in determining the robot's heading. An additional sensor is required to accurately measure heading as the integrated GPS and encoder data is insufficient. An inertial measurement unit (IMU) and a magnetometer are two options for an additional sensor. An IMU, though quite sensitive to heading changes, suffers from error accumulation; any error in a single measurement is carried forward into future measurements. A magnetometer may be less sensitive to minor course changes but does not suffer from the same error accumulation. The encoders organic to the P3-AT robot could be integrated with a magnetometer to help increase measurement sensitivity. More accurately determining the robot's heading would result in increased performance of the system.

Additional future work could include the incorporation of sensors such as a light detection and ranging (LIDAR) system and utilizing the SlimPRO SP675P's wireless communications capability to engage in swarm tactics with other Pioneer robots.

APPENDIX A. NAVIGATION PLATFORM

MATLAB code to provide control inputs to P3-AT robot based on GPS, encoder, and acoustic data:

```
clear all
clc
format long

%% first connect to the robot, 'Com1'
p3_connector('Com1');
disp('Connected to Robot')
pause(2)

%% Setup GPS Serial
s3 = serial('COM6','BaudRate', 4800,'InputBufferSize',128);
fopen(s3);   %opens the COM port
disp('GPS OPEN')
pause(1)

%% User input for goal
inputstring = ('Please enter goal coordinates');
disp(inputstring)

goal_coord = clipboard('paste'); %input goal pasted from KML clipboard
and GoogleEarth

lat_int = str2double(goal_coord(1:2));
lat_min = str2double(goal_coord(4:5));
lat_sec = str2double(goal_coord(7:11));
lat_dir = goal_coord(14);

lon_int = str2double(goal_coord(16:18));
lon_min = str2double(goal_coord(20:21));
lon_sec = str2double(goal_coord(23:27));
lon_dir = goal_coord(30);

lat_sec_dec = lat_sec/60;
lat_min_dec = (lat_min+lat_sec_dec)/60;
lat_dec = lat_int + lat_min_dec;
if lat_dir == 'S'
    lat_dec = -1*lat_dec;
end

lon_sec_dec = lon_sec/60;
lon_min_dec = (lon_min+lon_sec_dec)/60;
lon_dec = lon_int + lon_min_dec;
if lon_dir == 'W'
    lon_dec = -1*lon_dec;
end

LatGoal_coordinate = lat_dec;
```

```
LonGoal_coordinate = lon_dec;

Goal_coordinates = [LatGoal_coordinate,LonGoal_coordinate]

pause(2)

%% initialize heading in GPS
% p3_setTransVel(500)
% pause(5)
% p3_stopRobot
% disp('GPS Input Initialized')

%% Get initial GPS fix
GPGLL_flag = 0;

while GPGLL_flag == 0
    A1=fscanf(s3); %read serial port of GPS
    GPGLL_flag = strcmp(A1(1:6),'$GPGLL'); %pick off the GPGLL String
end

%pull lat and lon from GPGLL string and put into useable form
disp('GPS Data read')
a = mat2str(A1);
a_delim = strsplit(a,',');
lat_lon_cell = a_delim(2:5);
lat_lon_str = cell2mat(lat_lon_cell);
lat_str = lat_lon_str(1:9);
lat_dir = lat_lon_str(10);
lon_str = lat_lon_str(11:20);
lon_dir = lat_lon_str(21);

lat = str2double(lat_str);
lon = str2double(lon_str);
lat = lat/100;
lon = lon/100;

lat_int = floor(lat);
lat_dec = lat-lat_int;
lon_int = floor(lon);
lon_dec = lon-lon_int;

lat_dec_deg = lat_dec*100;
lon_dec_deg = lon_dec*100;

lat_true_decimal = lat_dec_deg/60;
lon_true_decimal = lon_dec_deg/60;

lat = lat_int+lat_true_decimal;
lon = lon_int+lon_true_decimal;
if lat_dir == 'S'
    lat = -1*lat;
end
```

```
if lon_dir == 'W'
    lon = -1*lon;
end

GPVTG_flag = 0;

while GPVTG_flag == 0
    B1=fscanf(s3); %read serial port of GPS
    GPVTG_flag = strcmp(B1(1:6),'$GPVTG'); %pick off the GPVTG String
end

b1 = mat2str(B1);
b_delim_i = strsplit(b1,',');
b_delim_ii = b_delim_i(2);
heading_str_i = cell2mat(b_delim_ii);

heading_i = str2double(heading_str_i);

disp('Initial fix and heading complete')


%% Compare fix and user input
FIX = [lat, lon];
HEADING = [heading_i];
checkhead = heading_i;

%% Initialize data logs
DIST = [];
sonarRanges_log = [];
q_log = [];
RobHead_log = [];
qGoal_log = [];
reqheading_log = [];

%% Required Constants
N = 1;
dist = 100;
RHO = 500;      % use with attractive force, eq. 2
ZETA =  3;     % use with attractive force. eq. 2
dC = 3000;     % cut-off distance  eq. 4
ETA = 1e6;     % constant coefficient eq. 4 was 1e6
sensX = .15;  % sensitivity for transVel
sensY = 15;  % sensitivity for rotVel

gamma = [90 50 30 10 -10 -30 -50 -90 -90 -130 -150 -170 170 150 130
90];
gamma = pi/180 * gamma;  % convert to rad

sonarLoc = 1e3*[ 0.069 0.136  ; %locations from P3DX vice P#AT but
differences minimal
                 0.114 0.119  ;
                 0.148 0.078  ;
                 0.166 0.027  ;
```

```
                 0.166 -0.027 ;
                 0.148 -0.078 ;
                 0.114 -0.119 ;
                 0.069 -0.136 ;
                -0.157 -0.136 ;
                -0.203 -0.119 ;
                -0.237 -0.078 ;
                -0.255 -0.027 ;
                -0.255 0.027  ;
                -0.237 0.078  ;
                -0.203 0.119  ;
                -0.157 0.136 ];


%% GPS Navigation
while dist >= 2 %continue to run control loop until less than 2 meters
from goal

%calculate position
GPGLL_flag = 0;

while GPGLL_flag == 0
    A2=fscanf(s3); %read serial port of GPS
    if length(A2) < 10
        pause(2)
        A2=fscanf(s3);
    end
    GPGLL_flag = strcmp(A2(1:6),'$GPGLL'); %pick off the GPGLL String
end

a = mat2str(A2);
a_delim = strsplit(a,',');
lat_lon_cell = a_delim(2:5);
lat_lon_str = cell2mat(lat_lon_cell);
lat_str = lat_lon_str(1:9);
lat_dir = lat_lon_str(10);
lon_str = lat_lon_str(11:20);
lon_dir = lat_lon_str(21);

lat = str2double(lat_str);
lon = str2double(lon_str);
lat = lat/100;
lon = lon/100;

lat_int = floor(lat);
lat_dec = lat-lat_int;
lon_int = floor(lon);
lon_dec = lon-lon_int;

lat_dec_deg = lat_dec*100;
lon_dec_deg = lon_dec*100;

lat_true_decimal = lat_dec_deg/60;
lon_true_decimal = lon_dec_deg/60;
```

```
lat = lat_int+lat_true_decimal;
lon = lon_int+lon_true_decimal;
if lat_dir == 'S'
    lat = -1*lat;
end

if lon_dir == 'W'
    lon = -1*lon;
end

FIX = [FIX;
    lat, lon];

clear A2 %clear buffer to ensure latest data being used next time
through control loop

GPVTG_flag = 0;

while GPVTG_flag == 0
    B=fscanf(s3); %read serial port of GPS
    GPVTG_flag = strcmp(B(1:6),'$GPVTG'); %pick off the GPVTG String
end

b = mat2str(B);
b_delim = strsplit(b,',');
b_delim2 = b_delim(2);
heading_str = cell2mat(b_delim2);

heading = str2double(heading_str);
HEADING = [HEADING;heading];

clear B  %clear buffer to ensure latest data being used next time
through control loop
%calculate required corrections/actions

latrad = deg2rad(lat);
lonrad = deg2rad(lon);
latradg = deg2rad(LatGoal_coordinate);
lonradg = deg2rad(LonGoal_coordinate);
deltalat = latradg-latrad;
deltalon = lonradg-lonrad;

%haversine functions
%required distance - determines while loop
dista = sin(deltalat/2)^2 + cos(latrad)*cos(latradg)*sin(deltalon/2)^2;
distc = 2*atan2(sqrt(dista),sqrt(1-dista));
dist = 6371000*distc %6.371E6 is radius of the earth in memters

DIST = [DIST;dist];
%required heading
```

```
goalheading =
atan2(sin(deltalon)*cos(latradg),cos(latrad)*sin(latradg)-
sin(latrad)*cos(latradg)*cos(deltalon));
goalheading = rad2deg(goalheading);
if goalheading < 0
    goalheading = goalheading + 360;
end

RobHead = p3_getXYHeading;
RobHead_log = [RobHead_log;RobHead];

%get organic DR heading from robot
if RobHead(3) >= 0
    RobHead3 = 360 - RobHead(3);
else
    RobHead3 = -1*RobHead(3);
end


% ensure heading accuracy, if GPS heading doesn't update utilize
organic DR
% heading from robot
if HEADING(N) == HEADING(N+1)
    heading_f = RobHead3 + checkhead;
else
    heading_f = heading;
end

%calculate the required change in heading (from current heading to goal
%heading)
reqheading = goalheading-heading_f;
if reqheading > 180
    reqheading = reqheading - 360;
elseif reqheading < -180
    reqheading = reqheading + 360;
end

reqheading_log = [reqheading_log;reqheading];


% shift goal to robot ref frame
xA = dist*cosd(reqheading);
yA = -1*dist*sind(reqheading);
RobHead1 = RobHead(1)/1000;
RobHead2 = RobHead(2)/1000;
rotA = [cosd(RobHead3),-sind(RobHead3), RobHead1;
        sind(RobHead3), cosd(RobHead3), RobHead2;
        0, 0, 1];

robgoal = rotA*[xA;yA;1];
xB = robgoal(1)*1000;
yB = robgoal(2)*1000;
qGoal = [xB;yB]; %goal in robot ref frame
```

```
qGoal_log = [qGoal_log; qGoal'];



sonarRanges = p3_getSonarData;
sonarRanges_log = [sonarRanges_log;sonarRanges];

q = [RobHead(1); RobHead(2)];  % actual robot x, y position
q_log = [q_log; q'];
theta = (RobHead3) * pi/180;   % robot heading in rads

% compute the ATTRACTIVE FORCE in Robot-coords
if(norm(q-qGoal) <= RHO)
    Fatt_w = -ZETA*(q-qGoal);
else
    Fatt_w = -ZETA*RHO * (q-qGoal)/norm(q-qGoal);
end
% transform Fatt_w to Fatt_r
Tw2r = [cos(theta) sin(theta); -sin(theta) cos(theta)];
Fatt_r = Tw2r * Fatt_w;
% compute the repulsive force
Frep_r = [0;0];
for ix = 1:16
    if(sonarRanges(ix) <= dC)
        Rs2r = [cos(gamma(ix))   -sin(gamma(ix));  % sensor to robot
frame
            sin(gamma(ix))      cos(gamma(ix))];
        ni = Rs2r * [sonarRanges(ix);0]  + [sonarLoc(ix,1);
sonarLoc(ix,2)];
        di = sonarRanges(ix);
        Frep_r = -ETA * (1/di - 1/dC)*ni./di + Frep_r;
    end
end
% command the robot motion
Ftotal_r = Frep_r + Fatt_r;
fwdVel = sensX * Ftotal_r(1);
rotVel = sensY * atan2(Ftotal_r(2),Ftotal_r(1));
p3_setTransVel(fwdVel);    % move forward
p3_setRotVel(rotVel);      % rotate

% wait a little bit for robot to catch up with Matlab
flushinput(s3)
N = N+1 %display loop number to ensure program is running
pause(1);
end

p3_stopRobot

flushinput(s3) %clear buffer to avoid fclose bug
fclose(s3) %close GPS port
p3_disconnector
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. DATA PROCESSING

MATALB code to process and plot data from trial runs:

```
figure(1)
plot(-qlog1(:,2),qlog1(:,1),'Linewidth',2);
hold on
scatter(-qGoal_log1(2:end,2),qGoal_log1(2:end,1))
scatter(-qGoal_log1(2,2),qGoal_log1(2,1),'filled')
scatter(-
qGoal_log1(length(qGoal_log1),2),qGoal_log1(length(qGoal_log1),1),'fill
ed')
legend('Robot Path','Calculated Goal','Initial Goal','Final
Goal','Location','NW')
title('Run 1')
xlabel('Dist (mm)')
ylabel('Dist (mm)')
hold off

figure(2)
plot(-qlog2(:,2),qlog2(:,1),'Linewidth',2);
hold on
scatter(-qGoal_log2(2:end,2),qGoal_log2(2:end,1))
scatter(-qGoal_log2(2,2),qGoal_log2(2,1),'filled')
scatter(-
qGoal_log2(length(qGoal_log2),2),qGoal_log2(length(qGoal_log2),1),'fill
ed')
legend('Robot Path','Calculated Goal','Initial Goal','Final
Goal','Location','NW')
title('Run 2')
xlabel('Dist (mm)')
ylabel('Dist (mm)')
hold off

figure(3)
plot(-qlog3(:,2),qlog3(:,1),'Linewidth',2);
hold on
scatter(-qGoal_log3(:,2),qGoal_log3(:,1))
scatter(-qGoal_log3(2,2),qGoal_log3(2,1),'filled')
scatter(-
qGoal_log3(length(qGoal_log3),2),qGoal_log3(length(qGoal_log3),1),'fill
ed')
legend('Robot Path','Calculated Goal','Initial Goal','Final
Goal','Location','NW')
title('Run 3')
xlabel('Dist (mm)')
ylabel('Dist (mm)')
axis([-6000 18000 0 25000])
hold off

figure(4)
plot(1:length(DIST1),DIST1,'Color','b','Linewidth',2)
```

```
hold on
plot(1:length(DIST2),DIST2,'Color','g','Linewidth',2)
plot(1:length(DIST3),DIST3,'Color','r','Linewidth',2)
title('Dist To Goal')
xlabel('Interations Through Control Loop')
legend('Run 1','Run 2','Run 3','Location','NE')
ylabel('Dist (meters)')
hold off

figure(5)
subplot(1,2,1)
plot(-qlog1(:,2),qlog1(:,1),'Linewidth',2);
hold on
scatter(-qGoal_log1(2:end,2),qGoal_log1(2:end,1))
scatter(-qGoal_log1(2,2),qGoal_log1(2,1),'filled')
scatter(-
qGoal_log1(length(qGoal_log1),2),qGoal_log1(length(qGoal_log1),1),'fill
ed')
legend('Robot Path','Calculated Goal','Initial Goal','Final
Goal','Location','NW')
title('Run 1')
xlabel('Dist (mm)')
ylabel('Dist (mm)')
axis([-6000 10000 0 25000])
hold off
subplot(1,2,2)
plot(-qlog2(:,2),qlog2(:,1),'Linewidth',2);
hold on
scatter(-qGoal_log2(2:end,2),qGoal_log2(2:end,1))
scatter(-qGoal_log2(2,2),qGoal_log2(2,1),'filled')
scatter(-
qGoal_log2(length(qGoal_log2),2),qGoal_log2(length(qGoal_log2),1),'fill
ed')
legend('Robot Path','Calculated Goal','Initial Goal','Final
Goal','Location','NW')
title('Run 2')
xlabel('Dist (mm)')
ylabel('Dist (mm)')
axis([-6000 10000 0 25000])
hold off

Init_Final_1 = sqrt((qGoallog1(1,1)-
qGoallog1(length(qGoallog1),1))^2+(qGoallog1(1,2)-
qGoallog1(length(qGoallog1),2))^2);
Init_Final_2 = sqrt((qGoallog2(1,1)-
qGoallog2(length(qGoallog2),1))^2+(qGoallog2(1,2)-
qGoallog2(length(qGoallog2),2))^2);
Init_Final_3 = sqrt((qGoallog3(1,1)-
qGoallog3(length(qGoallog3),1))^2+(qGoallog3(1,2)-
qGoallog3(length(qGoallog3),2))^2);

Dist_Goal_1 = sqrt((qGoallog1(length(qGoallog1),1)-
qlog1(length(qlog1),1))^2+(qGoallog1(length(qGoallog1),2)-
qlog1(length(qlog1),2))^2);
```

```
Dist_Goal_2 = sqrt((qGoallog2(length(qGoallog2),1)-
qlog2(length(qlog2),1))^2+(qGoallog2(length(qGoallog2),2)-
qlog2(length(qlog2),2))^2);
Dist_Goal_3 = sqrt((qGoallog3(length(qGoallog3),1)-
qlog3(length(qlog3),1))^2+(qGoallog3(length(qGoallog3),2)-
qlog3(length(qlog3),2))^2);
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

[1] "*Course Description and Outline*," class notes for EC4310 Fundamentals of Robotics, Dept of Electrical and Computer Engineering, Naval Postgraduate School, Monterey, CA, Fall Quarter Academic Year 2016.

[2] J. J. Craig, *Introduction to Robotics*, 2nd ed. Reading, MA: Addison-Wesley Publ. Co., 1989, pp. 1–59.

[3] D. G. Alciatore and M. B. Histand, *Introduction to Mechatronics and Measurement Systems*, 4th ed. New York, NY: McGraw-Hill, 2012, pp. 383-391.

[4] "*Sonar Sensors and Mapping with the Pioneer P3-DX Mobile Robot, Lab 2*" Class notes for EC4310 Fundamentals of Robotics, Dept. of Electrical and Computer Engineering, Naval Postgraduate School, Monterey, CA, Fall Quarter 2017.

[5] J. Latombe, *Robot Motion Planning*. Boston: Kluwer Academic Publishers, 1991, pp. 295–352.

[6] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins, *GPS: Theory and Practice*, 5th ed. Austria: SpringerWien. 2001, pp. 1–37.

[7] GPS control segment. (n.d.). U. S. government. [Online], Available: http://www.gps.gov/systems/gps/control/, Accessed: Nov 16, 2016.

[8] *Standard for Interfacing Marine Electronic Devices*, NMEA Standard 0183, 1994.

[9] R. W. Sinnott, "Virtues of the haversine," *Sky & Telescope*, vol 68, no. 2, p. 158, Dec 1984.

[10] *Global Positioning System Standard Positioning Service Performance Standard*, 4th ed. Washington, DC: Department of Defense, 2008.

[11] *BU-354-S4 USB GPS FAQ*, US Globalsat, Chino, CA. Available: http://usglobalsat.com/store/gpsfacts/bu353s4_gps_facts.html Accessed: Nov 20, 2016.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California